

An approach to secure weather and climate models against hardware faults

Peter D. Düben^{1,2}, Andrew Dawson¹

¹AOPP, Department of Physics, University of Oxford, Oxford, UK.

²European Centre for Medium Range Weather Forecasts, Reading, UK.

Key Points:

- An approach to secure dynamical cores of Earth System models against hardware faults is presented.
- Projections of prognostic parameters onto a coarse backup grid are used to secure model simulations.
- Runs with a shallow water model can be stabilized in the presence of hardware faults and bit flips.

Corresponding author: Peter D. Düben, peter.dueben@physics.ox.ac.uk

This article has been accepted for publication and undergone full peer review but has not been through the copyediting, typesetting, pagination and proofreading process which may lead to differences between this version and the Version of Record. Please cite this article as an 'Accepted Article', doi: 10.1002/2016MS000816

Abstract

Enabling Earth System models to run efficiently on future supercomputers is a serious challenge for model development. Many publications study efficient parallelisation to allow better scaling of performance on an increasing number of computing cores. However, one of the most alarming threats for weather and climate predictions on future high performance computing architectures is widely ignored: the presence of hardware faults that will frequently hit large applications as we approach exascale supercomputing. Changes in the structure of weather and climate models that would allow them to be resilient against hardware faults are hardly discussed in the model development community.

In this paper, we present an approach to secure the dynamical core of weather and climate models against hardware faults using a backup system that stores coarse resolution copies of prognostic variables. Frequent checks of the model fields on the backup grid allow the detection of severe hardware faults, and prognostic variables that are changed by hardware faults on the model grid can be restored from the backup grid to continue model simulations with no significant delay.

To justify the approach, we perform model simulations with a C-grid shallow water model in the presence of frequent hardware faults. As long as the backup system is used, simulations do not crash and a high level of model quality can be maintained. The overhead due to the backup system is reasonable and additional storage requirements are small. Runtime is increased by only 13% for the shallow water model.

1 Introduction

Weather and climate models are running on some of the fastest supercomputers in the world and Earth System modelling is a field of research that depends heavily on high performance computing (HPC). Until recently, the community of weather and climate modelling could always trust in an ever-increasing amount of computational power. Only a limited adjustment of model code was needed to port a model to run efficiently on state-of-the-art supercomputers. However, this is changing now as silicon technology is reaching physical limits in terms of the minimal size of transistors and energy density such that we cannot expect single processing cores to become much faster in the future. To counteract this stagnation in performance increase, future HPC computing hardware will use an ever increasing number of processing cores that compute HPC applications in parallel. To enable models running efficiently on a large number of computing cores in parallel is a serious challenge for the development of Earth System models and it becomes even more challenging when accelerators are used, such as GPUs (see for example *Fuhrer et al.* [2014]) and Intel's Knights Landing. Already today, huge efforts are made to scale atmosphere models to run efficiently on hundreds of thousands of processing cores in parallel on petascale supercomputers (see for example *Dennis et al.* [2012]; *Wyszogrodzki et al.* [2012]; *Johnsen et al.* [2013]; *Müller et al.* [2015]). However, the World's fastest supercomputer, the Sunway TaihuLight, is already running more than 10 million processing cores in parallel. If weather and climate models are to continue running efficiently on supercomputers in the future, requirements for HPC need to be taken seriously in the community of Earth System modelling to keep up with this development. On the one hand, models need to be designed to support parallelisation that is sufficient to keep all processing cores busy. On the other hand, communication and data needs to be reduced to a minimum since the performance of weather and climate models is typically limited by the availability of data, e.g. due to limited capacity and speed of memory and cache. The frequency of data processing, e.g. in floating point operations, is typically not the bottleneck. This implies that local computation will not necessarily slow down model simulations.

One of the most alarming properties of future HPC has been widely ignored in model development so far: future massively parallel supercomputers will suffer from fre-

quent hardware faults that will influence model simulations on various different levels. If millions of processing cores are run in parallel, it is likely that hardware will break frequently during a simulation, or that individual hardware components will be faulty and cause permanent hardware faults that will be very hard to detect, for example due to non-functional memory cells. Furthermore, soft errors, that manifest themselves in form of bit flips, can either cause the application to terminate immediately or cause a silent error that alters a simulation without causing the simulation to fail, resulting in an unpredictable impact on the quality of results. Frequent hardware failure is already reality for existing supercomputers. For example, the Los Alamos National Laboratory's ASCI Q machine experienced 27.7 CPU failures per week early in the deployment [Michalak *et al.*, 2005] and a Blue Gene/L at Livermore National Laboratory with a total of 65,536 nodes experienced faults at a rate of 48 hours during initial deployment [Wang *et al.*, 2007].

Making HPC hardware resilient against hardware faults is a huge effort in hardware development, and methods to increase resiliency are studied at almost all levels of hardware design (see for example Cappello *et al.* [2009] for an overview). However, it is unlikely that hardware development will be able to guarantee that hardware faults will not affect large scale HPC applications in the future. On the other hand, if a small rate of hardware faults can be accepted in large HPC applications, computing cost can be reduced significantly, since the requirement to produce the correct result under all circumstances requires a large amount of resources. The trade to reduce computing cost and allow a certain amount of faults is, for example, realized in stochastic processors that overscale the voltage which is applied to the processing core. This will cause occasional bit flips but allow significant energy savings (see for example Narayanan *et al.* [2010] and Sartori *et al.* [2011]). Results that have been published for simplified, prototype architectures show a reduction of power consumption of more than 10% at an error rate of 2% [Kahng *et al.*, 2010] and up to a multiplicative factor of 15 for the area-delay-energy product at relative error magnitudes of 10% [Lingamneni *et al.*, 2012]. However, given the premature state of this research it is very difficult to predict how such hardware would perform for floating point arithmetic in high performance computing.

One of the major research objectives to allow correct computation of massively parallel applications on exascale systems should be the investigation of fault oblivious algorithms that make the applications themselves resilient against hardware faults [Cappello *et al.*, 2009]. The easiest way to secure applications is to use check-pointing methods that store the state of a model simulation frequently to allow model restarts if faults cause an application to stop. However, simple check-pointing will generate very large overheads as the frequency of checkpoints needs to increase as frequency of faults increases with the number of processing cores. It is possible that the time to checkpoint and restart may exceed the mean time to be interrupted by faults in the future [Cappello *et al.*, 2009]. The necessity to make model simulations resilient against hardware faults is particularly pressing for simulations that are bound to strict deadlines and may not be able to be repeated in time if model simulations are terminated due to hardware faults, such as operational weather forecasts.

Some work has been done to diagnose the impact of hardware faults on HPC applications (for example Shantharam *et al.* [2011]), to detect soft errors in computations (for example Sloan *et al.* [2013]; Sharma *et al.* [2015]; Benson *et al.* [2015]), and to design numerical algorithms that are resilient against faults (for example Chen and Dongarra [2008]; Hoemmen and Heroux [2011]; Shantharam *et al.* [2012]; Cui *et al.* [2013]), mostly for linear solvers and linear operations such as matrix-matrix and matrix-vector multiplication. These are components that are certainly used within weather and climate models. However, the authors are not aware of any work that would suggest methods to make entire dynamical cores of weather and climate models resilient against hardware faults or study the impact of hardware faults and soft errors on simulations of weather and climate models. One exception is a study on the use of stochastic processors in atmospheric

modelling. Here, the use of stochastic processors is emulated in simulations of a three-dimensional spectral dynamical core (the Intermediate Global Climate Model (IGCM) or Reading Spectral Model). Results show that the model can cope with a significant number of bit flips in large parts of the model integration (e.g. error rates of 1 or 10% in floating point operations), but balancing problems perturbed simulations if bit flips were changing the most significant bits of the significand of floating point numbers (Düben *et al.* [2014], Düben and Palmer [2014]). However, in these studies bit flips were limited to the significand of floating point numbers and bit-flips for exponent or sign bits, that would have a much stronger impact on model simulations, were excluded.

In this paper, we suggest an approach to secure the dynamical core of weather and climate models against hardware faults. The scientific contribution of this paper is to show results that suggest that model simulations can be stabilised in the presence of frequent bit flips in floating point numbers, including flips in the sign and exponent bits, and that simulations can also be secured against hardware faults that wipe out all information on prognostic parameters within a significant area of the model domain (e.g. in case of the failure of an MPI tile). The method that is proposed to secure model simulations uses a backup copy of the model state on a coarser grid to make sure that additional data requirements are comparably small. To identify whether a model simulation was corrupted by hardware faults, frequent checks whether the model fields on the backup grid have changed by an unreasonable value are performed. If an unreasonable value was found, the corresponding field values of the model grid are also checked for whether they are reasonable and unreasonable values are replaced by projections from the backup to the model grid.

We do not claim that the backup system that is presented in this paper is the most efficient way to secure simulations against hardware faults. However, we do claim that we present a reasonable approach to allow model simulations at high quality in a supercomputing environment that shows occasional hardware faults. To the authors' best knowledge, this is the first approach to secure dynamical cores of Earth System models against hardware faults that does not require model restarts. It is the intention of this paper, to bring attention to the possibility to secure model simulations in fluid dynamics against hardware faults on a software level, to reduce requirement on the development of fault-free hardware that generates large cost overheads.

Section 2 introduces the backup system. Section 3 presents results for model simulations in the presence of hardware faults and provides estimates for the cost that is caused by the backup system. Section 4 explains limitations of the backup system. Section 5 presents the conclusions.

2 The backup system

In this section, we will present the backup system that detects severe hardware faults and replaces erroneous values of prognostic variables by approximations to continue simulations without delay. The main goal is to design a backup system for grid point models with the following properties:

- The system needs to enable stable simulations at high quality in the presence of hardware faults.
- Computational overhead needs to be as small as possible.
- The system should not influence model simulations when no hardware faults occur.
- The backup system itself should be resilient against hardware faults such that it can be run on the same hardware as the model.

To reduce the computational overhead due to the backup grid, and in particular the overhead in data volume that needs to be moved and stored, the backup system is based

on a grid at coarser resolution compared to the model grid. The backup system uses the following mechanism:

1. The prognostic variables from the model grid are mapped onto the backup grid of the backup system after model initialisation and at the end of each timestep. The values of the backup grid are kept in storage for one timestep, to allow a comparison with the model fields at the next timestep.
2. The fields on the backup grid are checked for the presence of “not a number” (NaN) values, and whether they have changed by an unexpected amount from one timestep to the next. The threshold of this check needs to be tuned to the specific model simulation under consideration. If the change of a model variable is larger than the threshold, it is assumed that a hardware fault has perturbed the simulation in either the model timestep on the model grid, or in the mapping between the model and the backup grid. Therefore, the specific value on the backup grid is replaced by the corresponding value from the previous timestep.
3. The corresponding values on the model grid, that influenced the erroneous grid value on the backup grid, are checked for NaNs or values outside of a physical meaningful range. This range needs to be adjusted to the model setup as well.
4. If the value on the model grid is found to be unreasonable, it is replaced by the value that is mapped from the backup grid onto the specific position of the model grid for the specific prognostic variable.

The entire model performs exactly the same way as the original model setup as long as no hardware faults are detected. For the mapping between the model and the backup grid, we need to take two constraints on the mapping procedure into account:

1. All grid point values of the model grid need to contribute to at least one grid point on the backup grid. If a specific grid parameter on the model grid would not contribute to any parameter on the backup grid, not all non-physical values on the model grid can be detected by checking the physical fields on the backup grid only. To check whether variable values are reasonable on the model grid would increase overheads.
2. Neighbouring grid points on the model grid need to contribute to degrees of freedom on the backup grid by a slightly different magnitude. If neighbouring grid points would have the same contribution to a grid point of the backup grid, this could cause problems if the flux between grid cells of a specific timestep was corrupted by a hardware fault. In this case, two neighbouring values will be perturbed by the same value with different sign and the error will level out when the backup grid value is calculated.

A backup system for a C-grid shallow water model

The model that is used for numerical tests in this paper is a standard implementation of the inviscid shallow water equations on an Arakawa C-grid which is based on the model used in *Marshall et al.* [2013]. The prognostic variables are surface elevation and velocity that follow the equations:

$$\partial_t u - (f + \bar{\zeta}^y) \bar{v}^{xy} + \partial_x B = 0, \quad (1)$$

$$\partial_t v - (f + \bar{\zeta}^x) \bar{u}^{xy} + \partial_y B = 0, \quad (2)$$

$$\partial_t h + \partial_x (\bar{H}^x u) + \partial_y (\bar{H}^y v) = 0, \quad (3)$$

with the Bernoulli Potential $B = gh + 0.5 \cdot (\bar{u}^x)^2 + 0.5 \cdot (\bar{v}^y)^2$, relative vorticity $\zeta = \partial_x v - \partial_y u$, the two components of velocity u and v , the Coriolis parameter f , the gravitational accel-

eration g , and the height of the fluid column H given by $H = h_0 + h - h_t$, where h_0 is the mean fluid depth, h is the surface elevation, and h_t is the topography. The overbars indicate averages in x or y direction on the C-grid to map corresponding field values between cell centres and cell edges. A third order Adams-Bashforth time-stepping scheme is used.

Figure 1. Sketches of the C-grid that is used for simulations. **a)** One grid cell of the backup grid (dark) and the corresponding degrees of freedom of the model grid (light). The position of the scalar h field is represented by dots in the cell centre, while zonal and meridional velocities are represented as arrows on the cell edges. **b)** Larger scale setup with many cells (arrows are not plotted). **c)** Coefficients that are used to map from the model to the backup grid cell. The sum of the coefficients within a backup grid cell is one. The same coefficients are used for the corresponding mapping procedure of the u and v field.

For the backup system that is used in this paper, we choose a backup grid spacing (Δx_b) that is three times larger compared to the grid spacing of the model grid (Δx_m ; see Figure 1). The C-grid setup allows simple relations between the model and the backup grid, as each grid point of the model grid has a nearest neighbour on the backup grid and each grid point on the backup grid has nine associated grid points on the model grid, including one grid point on the model grid that has the exact same location as a grid point on the backup grid. This is true for all prognostic variables (u, v, h) on the staggered grid. To map fields from the model grid to the backup grid, we implemented the mapping described in Figure 1. To map a field value from the backup to the model grid, we use a bilinear mapping. The backup mechanism will clearly only restore an approximation of the field value that was perturbed by the hardware fault. There are several sources of errors in this approximation such as the fact that the value on the backup grid that was used to restore the field value is from the previous timestep and represented on a much coarser grid, or the problem that a tendency that is stored for the Adams-Bashforth time-stepping scheme may be perturbed by a fault such that the same fault will probably appear in the following two timesteps. However, the backup system needs to trade precision against minimal overheads.

3 Numerical simulations

We present results of numerical simulations in this section. We introduce the test cases in Section 3.1, and show results for simulations with bit flips and hardware faults that simulate the loss of an entire MPI tile in Sections 3.2 and 3.3 respectively. The additional cost that is generated by the backup system is discussed in Section 3.4.

3.1 The test cases

The first testcase simulates flow over an isolated mountain that generates wave patterns in the lee of the mountain. The second testcase is using random numbers as initial conditions to study the two-dimensional turbulent cascade.

The isolated mountain testcase

We simulate zonal flow over an isolated mountain. Coriolis force is set to zero $f = 0.0$ for this testcase, since it would change the direction of the background flow over time. Simulations are performed in a rectangular domain ($L_x = 1,200,000\text{m}$ and $L_y = 200,000\text{m}$) with a background flow in x direction ($u_0 = 10\text{m/s}$ and $v_0 = 0\text{m/s}$). We use a timestep of two seconds and a mean fluid depth of $h_0 = 400\text{m}$. The model grid is using 180×60 grid cells. The topography is a mountain in the shape of a Gaussian:

$$h_t = 100.0 \cdot \exp\left(-\frac{(x - L_x/8)^2}{\sigma^2} - \frac{(y - L_y/2)^2}{\sigma^2}\right) \text{m}, \quad (4)$$

where $\sigma = \frac{3 \cdot L_y}{20}$. For this testcase, we define the physically meaningful range for the prognostic parameters that is needed by the backup system by $abs(h) < 8.0\text{m}$, $abs(u - 10.0) < 2.0\text{ms}^{-1}$ and $abs(v) < 1.0\text{ms}^{-1}$. We test for hardware faults if the absolute value of the corresponding prognostic field on the backup grid changes by $\Delta h > 0.05\text{m}$, $\Delta u > 0.01\text{ms}^{-1}$ or $\Delta v > 0.01\text{ms}^{-1}$ within one timestep.

The turbulent cascade testcase

The prognostic variables of the model simulation are initialised with uniform white noise in the range between -0.5m/s and 0.5m/s for u and v and -50m and 50m for h . This testcase is simulated on an f-plane with $f = 1.0 \cdot 10^{-4}\text{s}^{-1}$, no topography and $h_0 = 400\text{m}$. If simulations are started from white noise, eddies form and increase in size over time, as expected for the shallow water setup with a two-dimensional turbulent cascade. We initialise a simulation in double precision and run it for five million timesteps with a timestep of 25 seconds to obtain meaningful initial conditions for which eddies are still reasonably small. The size of the domain is $10,000\text{km} \times 10,000\text{km}$. The model grid is using 150×150 grid cells.

For this testcase, we define the physically meaningful range for the prognostic parameters that is needed by the backup system to be $abs(h) < 30.0\text{m}$, $abs(u) < 3.0\text{ms}^{-1}$ and $abs(v) < 3.0\text{ms}^{-1}$ and test for hardware faults if the absolute value of the prognostic fields change by $\Delta h > 1.0\text{m}$, $\Delta u > 0.5\text{ms}^{-1}$ or $\Delta v > 0.5\text{ms}^{-1}$ within one timestep.

3.2 Simulations with bits flips

Figure 2. Height field for the isolated mountain testcase calculated with random bit flips and fault rates of $p = 0.0$, $p = 10^{-9}$, $p = 10^{-7}$, $p = 10^{-6}$ (from top to bottom) after 1,000, 10,000 and 100,000 timesteps (from left to right).

Figure 3. Height field for the turbulent cascade testcase calculated with random bit flips and fault rates of $p = 0.0$, $p = 10^{-9}$, $p = 10^{-7}$, $p = 10^{-6}$ (from top to bottom) after 100,000, 500,000 and 1,000,000 timesteps (from left to right).

This section investigates the use of hardware that allows occasional bit flips in floating point arithmetic. These bit flips will happen in the entire model except for model initialisation and model output. The calculations of the backup system itself will also be performed on hardware that shows the same probability for bit flips. Bit flips will happen in all bits of floating point numbers – including exponent and sign – such that the impact of a single bit flip can be extremely strong (e.g., a floating point value can jump from 2m/s to $2.68 \cdot 10^{+154}\text{m/s}$ due to a single bit flip). We emulate the use of such a hardware within model simulations. Whenever a floating point operation is performed or whenever a value is assigned to a floating point number, the emulator will introduce bit flips to one of the 64 bits of the resulting floating point number at a prescribed fault rate. If, for example, $a = b + c \cdot d$ is calculated, the emulator will check and induce bit flips three times (indicated by the brackets in the following $a = [b + [c \cdot d]]$). For each check, the emulator

will perform a bit flip with the probability of p . If the emulator is introducing a bit flip, it will pick one of the 64 bits of the floating point number randomly and flip its value. The emulator that is used in this paper is based on an emulator for reduced numerical precision that is published on github [Dawson and Düben, 2016; Dawson and Düben, 2016].

Figure 2 and 3 show results of model simulations that use probabilities for bit flips of $p = 10^{-6}$, $p = 10^{-7}$, and $p = 10^{-9}$. For a simulation with a fault rate for all floating point operations of $p = 10^{-6}$, 50 bitflips per second would occur on an 1 GHz processor that is used at 5% of its peak performance (assignments are not considered for this estimate). It can be argued that these fault rates are much higher than fault rates that are expected for future hardware. However, since testcases that are studied in this paper are several orders of magnitude smaller when compared to high resolution weather and climate simulations, it is useful to study high fault rates to avoid being too optimistic with regard to model stability.

For $p = 10^{-6}$, simulations are clearly perturbed but the influence of bit flips reduces significantly as p is reduced and we see hardly any changes of results in simulations with $p = 10^{-9}$. We have not experienced any stability problems for fault rates up to $p = 10^{-6}$ for simulations that used the backup system. However, for simulations that were using the backup system at very high fault rates, we encountered unstable simulations. For example: A simulation with the turbulent cascade testcase at a fault rate of $p = 10^{-5}$ crashed after 36,077 timesteps. If no backup system is used, NaN values will be produced within 137, 214, and 47,659 timesteps for the isolated mountain and 9, 53, and 7,942 timesteps for the turbulent cascade testcase for $p = 10^{-6}$, $p = 10^{-7}$ and $p = 10^{-9}$ respectively. These numbers suggest that model simulations without a backup system would hardly allow useful simulations at the fault rates that are considered in this paper even if restart files are written frequently. There is a 11%, 1.7% and 0.0090% probability for the isolated mountain and a 46%, 4.7% and 0.042% probability for the turbulent cascade testcase to have a hardware fault within the first timestep that is severe enough to be detected by the backup system for the three fault rates respectively (probabilities are calculated from 100,000 runs).

3.3 Simulations with hardware faults

Figure 4. Area for which the information of prognostic parameters is wiped out when a hardware fault is emulated. Black: domain size; Green: loss of 6.26%; Blue: loss of 25%; Red: loss of 50%. The selected areas could be the area of the domain that is computed by a specific MPI task if 2, 4 or 16 MPI tasks are used in parallel to integrate the model.

Figure 5. Height field for the isolated mountain testcase calculated with hardware faults that wipe out prognostic variables in 0%, 6.25%, 25% and 50% of the area of the domain (from top to bottom) after 9,000, 49,000 and 99,000 timesteps, plotted after 10,000, 50,000 and 100,000 timesteps (from left to right).

Figure 6. Height field for the turbulent cascade testcase calculated with hardware faults that wipe out prognostic variables in 0%, 6.25%, 25% and 50% of the area of the domain (from top to bottom) after 99,000, 499,000 and 999,000 timesteps, plotted after 100,000, 500,000 and 1,000,000 timesteps (from left to right).

In this section, we consider hardware faults that wipe out information of prognostic variables in a large area of the domain. We simulate a hardware fault by setting all prognostic variables and their tendencies in the Adams Bashforth time stepping in a specific area to NaN from one timestep to the next. Such an error pattern could mimic a failure of an MPI tile during a simulation. To be able to recover information from the backup grid in a real hardware setup, it is assumed that the backup fields are not stored within the hardware component that failed. A second assumption on the hardware setup is that field values such as h_b can be restored quickly for calculations from memory or storage. Both assumptions will certainly produce computational overheads and will require solutions that are hardware specific and beyond the scope of this paper.

We perform simulations that lose information in 0%, 6.25%, 25% and 50% of the domain. Figure 4 shows the specific area for which information is lost in the simulations relative to the model domain. The number of degrees-of-freedom that are lost when the failure of an MPI tile is mimicked within simulations with the shallow water model in this paper may be comparable for heavily parallelised model simulations on exascale supercomputers in the future. However, the relative size of the area for which information is lost in comparison to the model domain is naturally much larger for simulations with 4-16 processors when compared to simulations with thousands of processors in supercomputing applications. Results of simulations that have three hardware faults within the run are shown in Figure 5 and 6. The model fields are plotted 1,000 timesteps after a hardware fault has happened. To study the worst case, hardware faults are happening three times at the same location within one simulation. In both test cases it is clearly visible that hardware faults have an influence on the model simulations. In the isolated mountain testcase, spurious waves develop for the 50% case. For the testcase with random initial conditions, it is visible that fine-scale features are lost, such that the backup system will influence the energy budget in case of a hardware fault. However, for both test cases, the impact of the hardware faults becomes small when the affected area is reduced with only small visible perturbations of the model simulation for the 6.25% simulations.

3.4 Numerical cost for the backup system

The backup system will generate only small overheads for data handling and data storage since the number of degrees of freedom for each model field is much smaller on the backup grid, compared to the model grid (a factor nine in the simulations of this paper; $\frac{\Delta x_b \cdot \Delta y_b}{\Delta x_m \cdot \Delta y_m} = 9$). Furthermore, 13 fields are stored for each grid point on the model grid (u, v, h , their tendencies of the previous timesteps for the Adams Bashforth scheme and h_b) compared to six floating point numbers for each grid point on the backup grid such that the data overhead due to the backup system is very small in comparison. The backup system requires checks that test whether parameters are reasonable and whether local replacements of parameters are required. However, since these tests are realised on the coarse backup grid, the computational overheads will be small, as long as the fault rate remains reasonably small such that the backup system does not need to replace prognostic parameters on the model grid frequently. The only calculation of the backup system that is comparably expensive is the mapping of the prognostic variables from the model to the backup grid. In this paper, this mapping is done every timestep. It is likely that it would be possible to reduce this rate to every second timesteps or less. However, this would allow hardware faults to propagate through the grid and increase the impact of individual hardware faults.

We can measure overheads due to the backup system that are caused when no hardware faults are happening in a standard double precision model simulation. For the model setup that was used in the previous sections for the turbulent cascade testcase, the use of the backup system increases compute time by 13%. This number was calculated for a serial programme on an Intel(R) Core(TM) i7-4770 CPU at 3.40 GHz using the GNU

Fortran compiler with optimisation level 3 and averaged over many simulations. Model initialisation and model input and output was not considered. Unfortunately, it is difficult to make credible estimates on how expensive the backup system will be if it is actively changing variables on the model grid. Simulations that use emulated bit flips cannot be evaluated in this perspective since the emulator for hardware faults introduces a significant computational overhead. However, simulations with the emulator at different fault rates and $p = 0.0$ show hardly any difference in integration time. Therefore, we expect the delay due to active parameter changes to be small.

Conventional methods to guarantee fault-resiliency that are already used in computing centres today, for example if new hardware is installed or if silent errors are anticipated, perform all model simulations twice or three times in parallel and compare results after every timestep to make sure that results are bitidentical. This will naturally increase computing cost by 100% or 200% respectively.

4 Discussion of limitations of the backup system

There are certainly limitations for the use of the proposed backup system. **The backup system will sacrifice bit-reproducibility.** Bit-reproducibility is considered to be important in operational weather forecasts since model simulation that crash should be reproducible to find the source of the problem. In climate modelling, bit reproducibility is a wanted feature since it allows to reproduce interesting weather phenomena, such as a tropical cyclone, for which not all data has been stored in the original simulation. However, the hardware that is considered in this paper, does not allow bit-reproducible simulations by definition. It is also likely that constraints on bit reproducibility will need to be softened in the future anyway since it will become more and more expensive to guarantee bit-reproducible simulations as models are run on more and more processing cores in parallel. Already today, computing power could be saved if model simulations were run with no guarantee for bit reproducibility. For example, when the order of operations is not guaranteed to be always the same when models are parallelised using MPI, savings can be of the order of 10%. However, rounding problems due to the order of operations can often be fixed by throwing away the least significant bits from global sums. Furthermore, the number of silent errors that will not crash the model but destroy bit-reproducibility will increase in the future and the increase in popularity of ensemble methods in both weather and climate predictions will, to some extent, reduce the need for bit-reproducible calculations.

The backup system is not able to distinguish between model errors and hardware errors. Therefore, it will prevent model crashes that are caused by the model itself, for example when using a model timestep that is too large. Whenever prognostic fields will leave the physical meaningful range of values that was predefined, the backup system will become active. This will make model debugging more complicated and it may be necessary to perform model testing with the backup system switched off.

The backup system will only detect severe hardware faults and bitflips in less significant bits or in less significant parts of model code will go undetected. This is due to the use of a check for the physically plausible range for the prognostic parameters. This range needs to be adjusted by the user and this adjustment may have a strong influence on the quality of model simulations in the presence of hardware faults. Therefore, a certain amount of tuning will be necessary to find the optimal configuration for the backup system.

The backup system will violate local mass conservation and global energy conservation. Therefore, restoring the state from the backup grid should be avoided where-ever possible. To this end, it may be sufficient for some time in the future to use the backup system solely to identify hardware faults and to restore the model state by re-running it

from the latest checkpoint. This may be a valid compromise between existing methods and the method proposed in this paper, as long as the time to checkpoint and restart does not exceed the mean time to be interrupted by faults. For low fault rates, it may well be possible to combine the error detection after every timestep suggested in this paper with a conventional checkpoint system at full resolution and precision in memory in intervals of a couple of timesteps, to recover the model state with no interpolation or conservation error whenever a hardware fault was detected.

The backup system does not guarantee stable model simulations and hardware faults can still cause model simulations to crash e.g. in the case of random bit flips that perturb both the present and the previous value of a specific prognostic variable on the backup grid and the corresponding prognostic variables on the model grid in the same time step. Bit flips that will impact instruction sets of hardware and the organisation of the model simulation as well as integer arithmetic are not addressed in this paper and the discussion of specific realisations of the backup system for a specific hardware setup, that will almost certainly reveal more challenges in the implementation, is beyond the scope of this paper.

The backup system will not detect model errors when calculating model diagnostics and model output. However, the calculation of diagnostics can cause a substantial fraction of computing cost in Earth System modelling. While errors in the diagnostic part of the model will go undetected in the configuration of the backup system as it is discussed in this paper, it may be possible to secure those parts with a similar backup system that is also based on the use of a coarser backup grid.

It can be argued that the use of ensemble simulations will reduce the need for error resilient methods in Earth System modelling. Ensemble methods gain more and more popularity since they improve estimates of model error and uncertainty for both weather and climate predictions. Since ensemble members are run in parallel there is no need to scale a single simulation to the size of the entire supercomputer. However, if hardware errors will cause a single simulation that is using a large fraction of the supercomputer to fail frequently, ensemble simulations that use the same fraction of the computer will likewise suffer from hardware faults with the difference that individual ensemble members will fail instead of the entire simulation. It will be difficult to perform reliable forecasts at a consistent level of quality if the number of ensemble simulations that finished successfully depends on the number and distribution of hardware faults.

5 Conclusion

This paper presents for the first time an approach to secure dynamical cores of weather and climate models against hardware faults that cause either random bit flips or a loss of information of prognostic parameters in a certain area of the model grid. It is likely to become essential to secure simulations of weather and climate models against hardware faults in the future as the number of processing cores that are used in parallel – and therefore the probability of hardware failures – is increasing rapidly. In particular for time critical applications such as operational weather forecasts for which the termination of a single simulation due to a hardware fault may have disastrous impact on the availability and quality of the entire weather forecast.

With all the limitations that were discussed in the previous section in mind, the authors still believe that this paper provides a significant scientific contribution towards weather and climate simulations on exascale supercomputers. We believe that developers of Earth System models will need to be open for hardware that cannot guarantee fault free double precision floating point arithmetic to be able to exploit future supercomputing hardware and to avoid a huge waste of resources spent to guarantee model simulations with no hardware faults. In this perspective, this paper is a first (incomplete) step with the

aim to widen the discussions about future model developments towards exascale super-computing with regards to error resilient model setups.

Acknowledgments

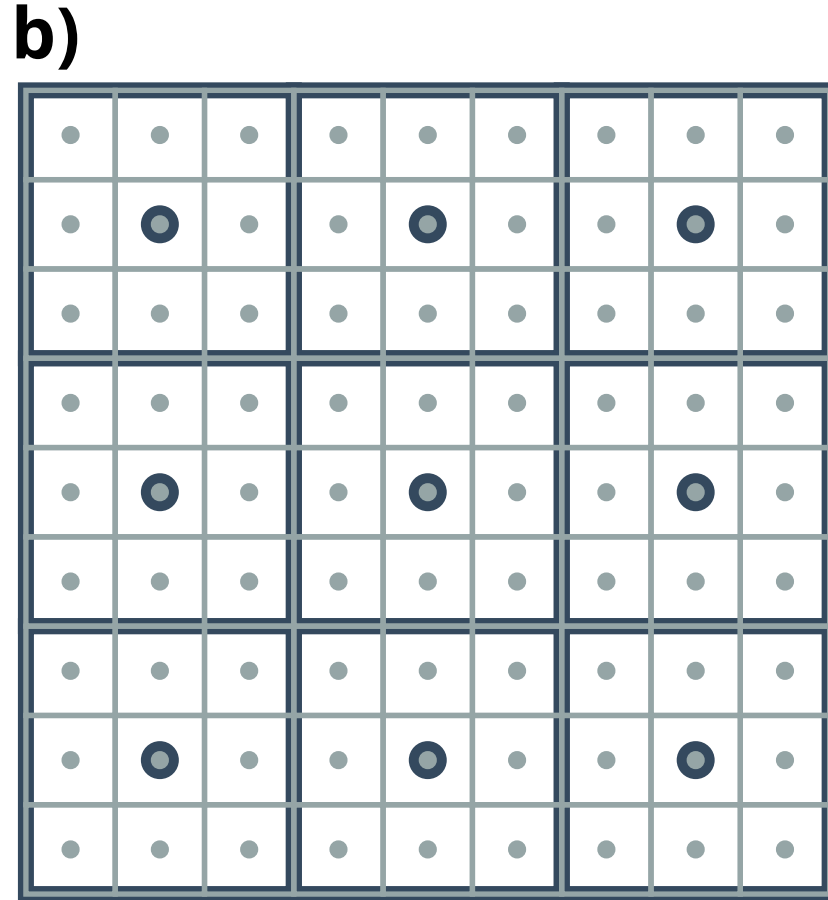
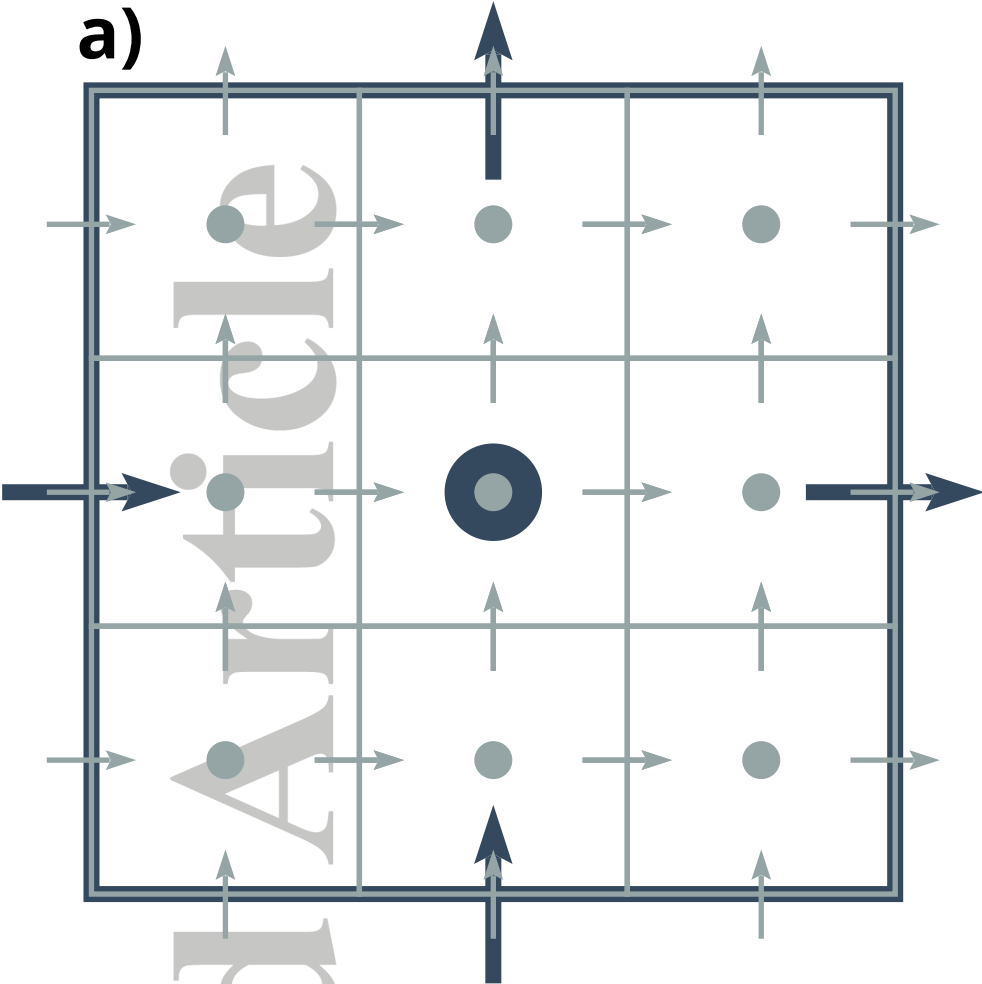
Many thanks to Tim Palmer, Fenwick Cooper, Nils Wedi, Colin Cotter and two reviewers for very helpful discussions and feedback. The authors received funding from an ERC grant (Towards the Prototype Probabilistic Earth-System Model for Climate Prediction, project reference 291406). No primary data is used for this publication. The code of the shallow water model is available at http://github.com/dueben/hardware_faults.

References

- Benson, A. R., S. Schmit, and R. Schreiber (2015), Silent error detection in numerical time-stepping schemes, *Int. J. High Perform. Comput. Appl.*, 29(4), 403–421.
- Cappello, F., A. Geist, B. Gropp, L. Kale, B. Kramer, and M. Snir (2009), Toward exascale resilience, *International Journal of High Performance Computing Applications*, 23(4), 374–388.
- Chen, Z., and J. Dongarra (2008), Algorithm-based fault tolerance for fail-stop failures, *IEEE Transactions on Parallel and Distributed Systems*, 19(12), 1628–1641.
- Cui, T., J. Xu, and C.-S. Zhang (2013), An error-resilient redundant subspace correction method., *CoRR*, *abs/1309.0212*.
- Dawson, A., and P. D. Düben (2016), An emulator for reduced floating-point precision written in fortran: <https://github.com/aopp-pred/rpe>.
- Dawson, A., and P. D. Düben (2016), rpe v5: An emulator for reduced floating-point precision in large numerical simulations, *Geoscientific Model Development Discussions*, 2016, 1–16, doi:10.5194/gmd-2016-247.
- Dennis, J. M., J. Edwards, K. J. Evans, O. Guba, P. H. Lauritzen, A. A. Mirin, A. St-Cyr, M. A. Taylor, and P. H. Worley (2012), Cam-se: A scalable spectral element dynamical core for the community atmosphere model, *International Journal of High Performance Computing Applications*, 26(1), 74–89.
- Düben, P. D., and T. N. Palmer (2014), Benchmark tests for numerical forecasts on inexact hardware, *Mon. Wea. Rev.*, 142, 3809–3829.
- Düben, P. D., H. McNamara, and T. N. Palmer (2014), The use of imprecise processing to improve accuracy in weather & climate prediction, *Journal of Computational Physics*, 271(0), 2–18, *Frontiers in Computational Physics Modeling the Earth System*.
- Fuhrer, O., C. Osuna, X. Lapillonne, T. Gysi, B. Cumming, M. Bianco, A. Arteaga, and T. Schulthess (2014), Towards a performance portable, architecture agnostic implementation strategy for weather and climate models, *Supercomputing frontiers and innovations*, 1(1).
- Hoemmen, M., and M. A. Heroux (2011), Fault tolerant iterative methods via selective reliability, in *Proceedings of the 2011 International Conference on High Performance Computing, Networking, Storage and Analysis*.
- Johnsen, P., M. Straka, M. Shapiro, A. Norton, and T. Galarneau (2013), Petascale wrf simulation of hurricane sandy deployment of ncsa’s cray xe6 blue waters, in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC ’13, pp. 63:1–63:7, ACM, New York, NY, USA.
- Kahng, A., S. Kang, R. Kumar, and J. Sartori (2010), Slack redistribution for graceful degradation under voltage overscaling, in *Design Automation Conference (ASP-DAC)*, 2010 15th Asia and South Pacific, pp. 825 –831.
- Lingamneni, A., K. K. Muntimadugu, C. Enz, R. M. Karp, K. V. Palem, and C. Piguet (2012), Algorithmic methodologies for ultra-efficient inexact architectures for sustaining technology scaling, in *Proceedings of the 9th conference on Computing Frontiers*, CF ’12, pp. 3–12, ACM, New York, NY, USA.

- Marshall, D. P., B. Vogel, and X. Zhai (2013), Rossby rip currents, *Geophysical Research Letters*, 40(16), 4333–4337.
- Michalak, S. E., K. W. Harris, N. W. Hengartner, B. E. Takala, and S. A. Wender (2005), Predicting the number of fatal soft errors in Los Alamos National Laboratory's asc q computer, *IEEE Transactions on Device and Materials Reliability*, 5, 329–335.
- Müller, A., M. A. Kopera, S. Marras, L. C. Wilcox, T. Isaac, and F. X. Giraldo (2015), Strong scaling for numerical weather prediction at petascale with the atmospheric model NUMA, *CoRR*, abs/1511.01561.
- Narayanan, S., J. Sartori, R. Kumar, and D. L. Jones: (2010), Scalable stochastic processors, in *proc. of Design, Automation and Test in Europe Conference*, pp. 335 – 338.
- Sartori, J., J. Sloan, and R. Kumar (2011), Stochastic computing: Embracing errors in architecture and design of processors and applications, in *Compilers, Architectures and Synthesis for Embedded Systems (CASES), 2011 Proceedings of the 14th International Conference on*, pp. 135 –144.
- Shantharam, M., S. Srinivasmurthy, and P. Raghavan (2011), Characterizing the impact of soft errors on iterative methods in scientific computing, in *Proceedings of the International Conference on Supercomputing*, ICS '11, pp. 152–161, ACM.
- Shantharam, M., S. Srinivasmurthy, and P. Raghavan (2012), Fault tolerant preconditioned conjugate gradient for sparse linear system solution, in *Proceedings of the 26th ACM International Conference on Supercomputing*, ICS '12, pp. 69–78, ACM, New York, NY, USA.
- Sharma, V., G. Gopalkrishnan, and G. Bronevetsky (2015), Detecting soft errors in stencil based computations, *LLNL-TR-670435*.
- Sloan, J., R. Kumar, and G. Bronevetsky (2013), An algorithmic approach to error localization and partial recomputation for low-overhead fault tolerance, in *2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 1–12.
- Wang, C., F. Mueller, C. Engelmann, and S. L. Scott (2007), A job pause service under lam/mpi+ blcr for transparent fault tolerance, in *21th International Parallel and Distributed Processing Symposium (IPDPS 2007), Proceedings*.
- Wyszogrodzki, A. A., Z. P. Piotrowski, and W. W. Grabowski (2012), *Parallel Processing and Applied Mathematics: 9th International Conference, PPAM 2011, Torun, Poland, September 11-14, 2011. Revised Selected Papers, Part II*, chap. Parallel Implementation and Scalability of Cloud Resolving EULAG Model, pp. 252–261, Springer Berlin Heidelberg, Berlin, Heidelberg.

Figure 1.



This article is protected by copyright. All rights reserved.

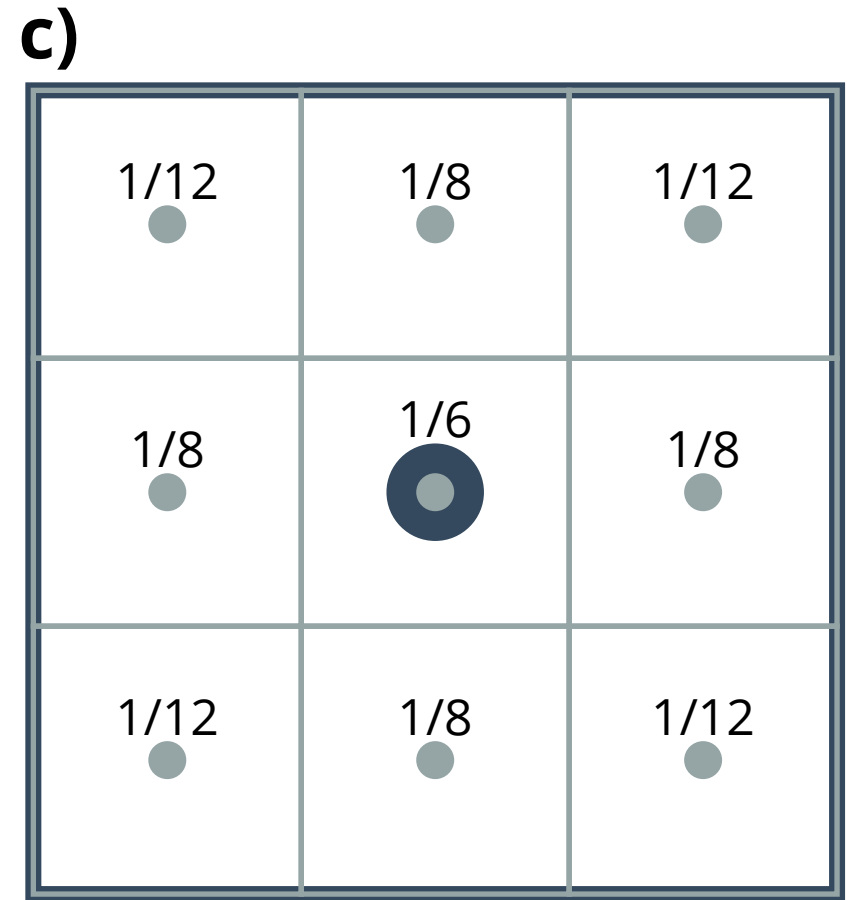


Figure 2.

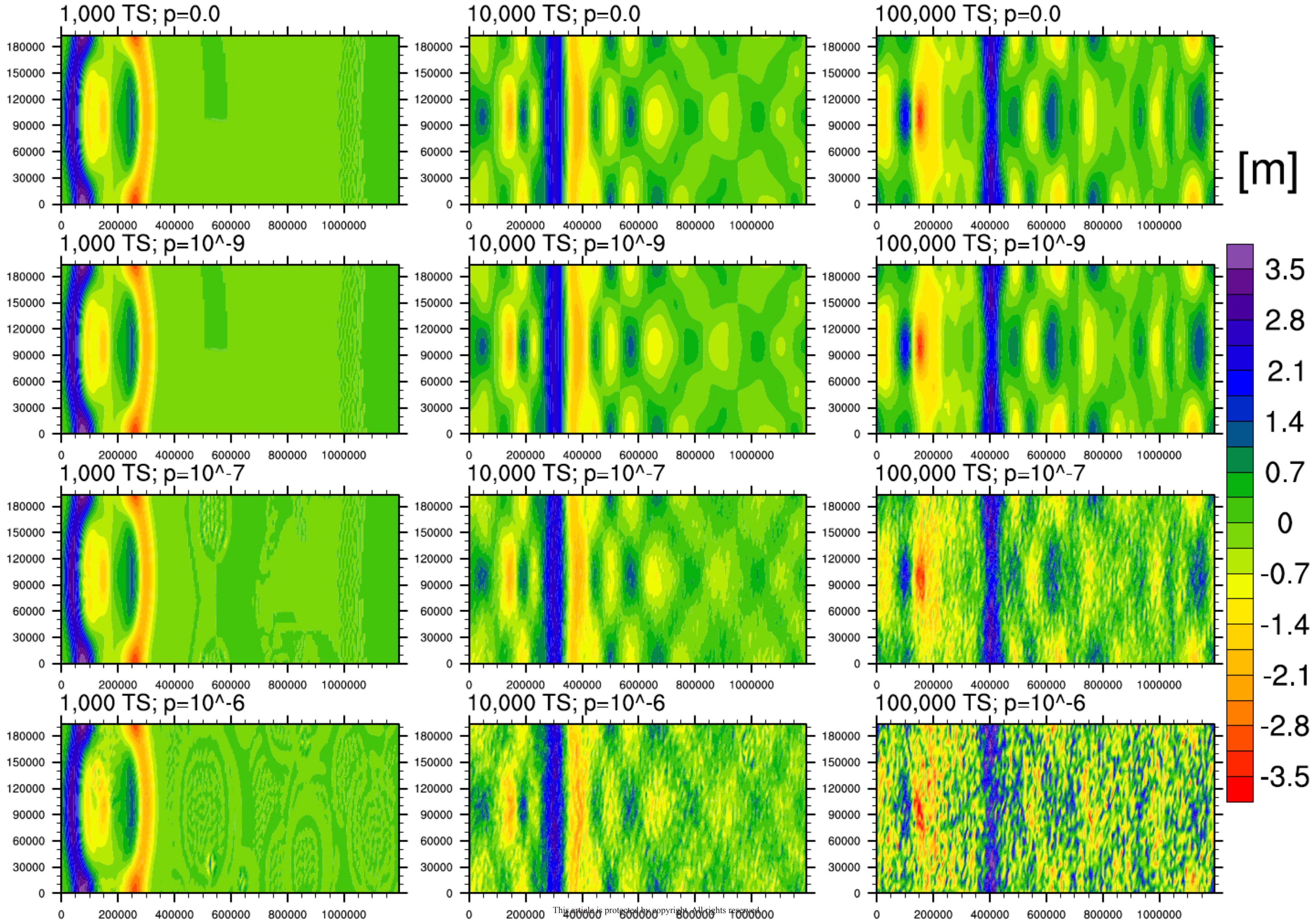


Figure 3.

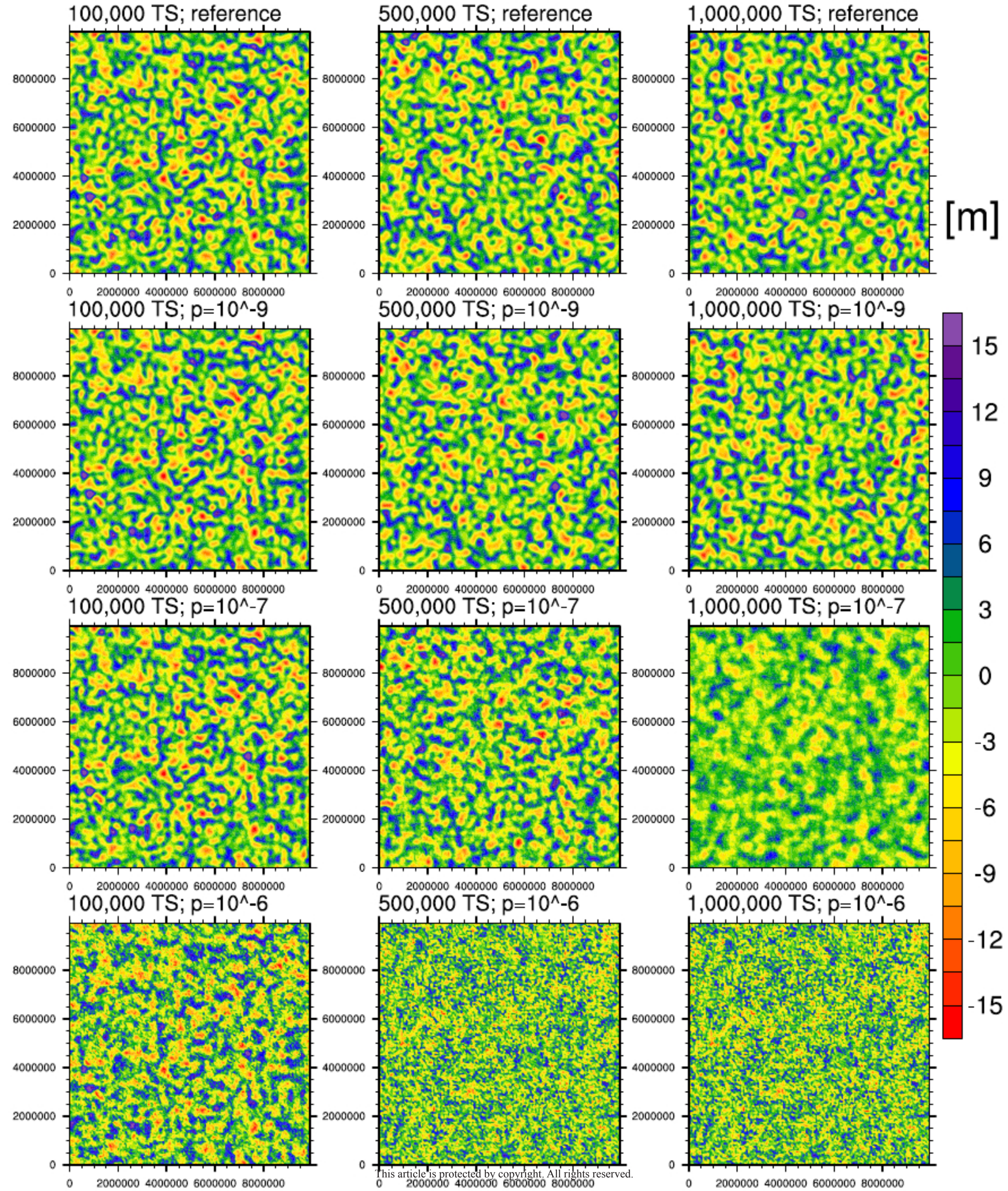


Figure 4.

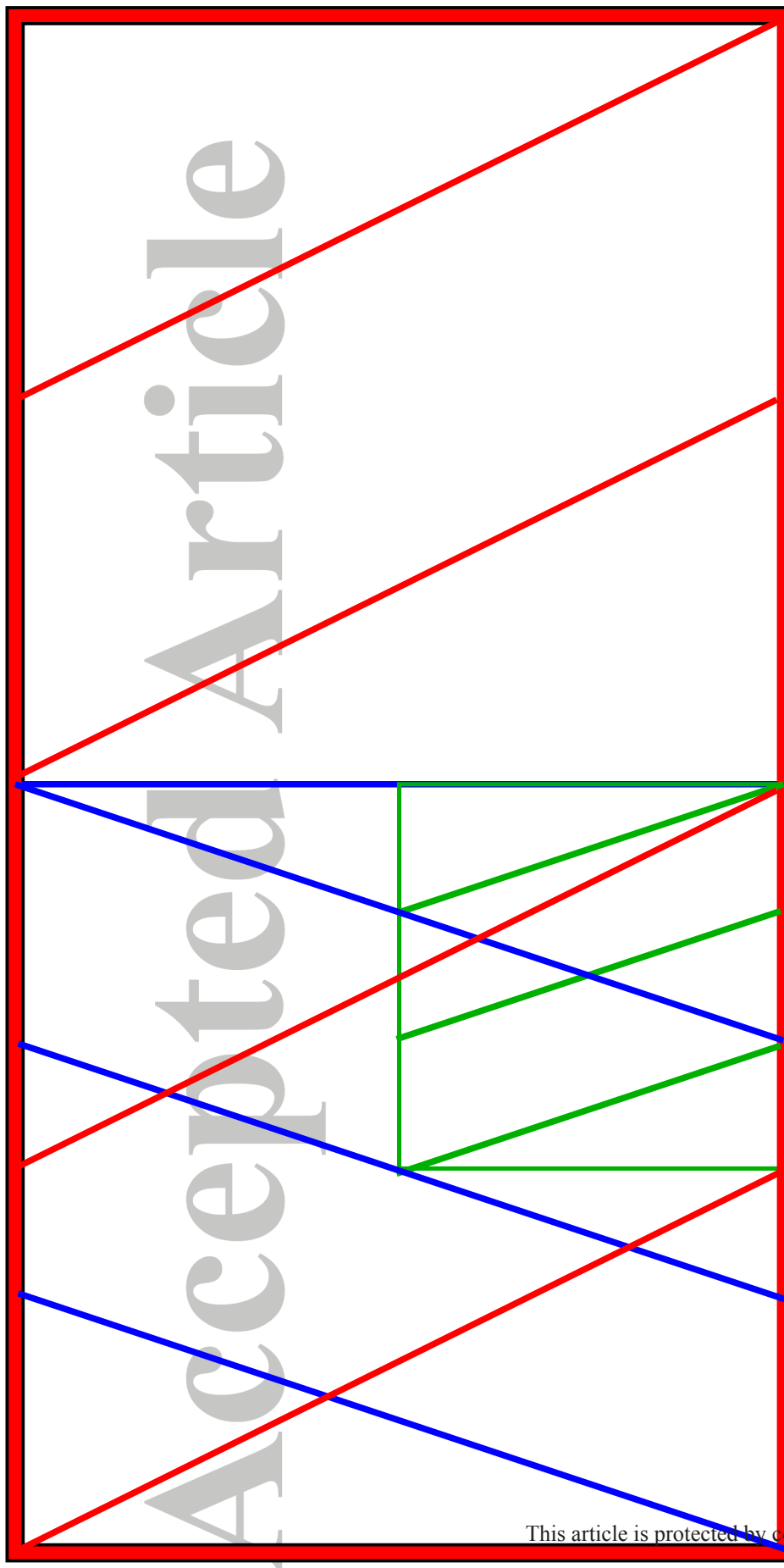


Figure 5.

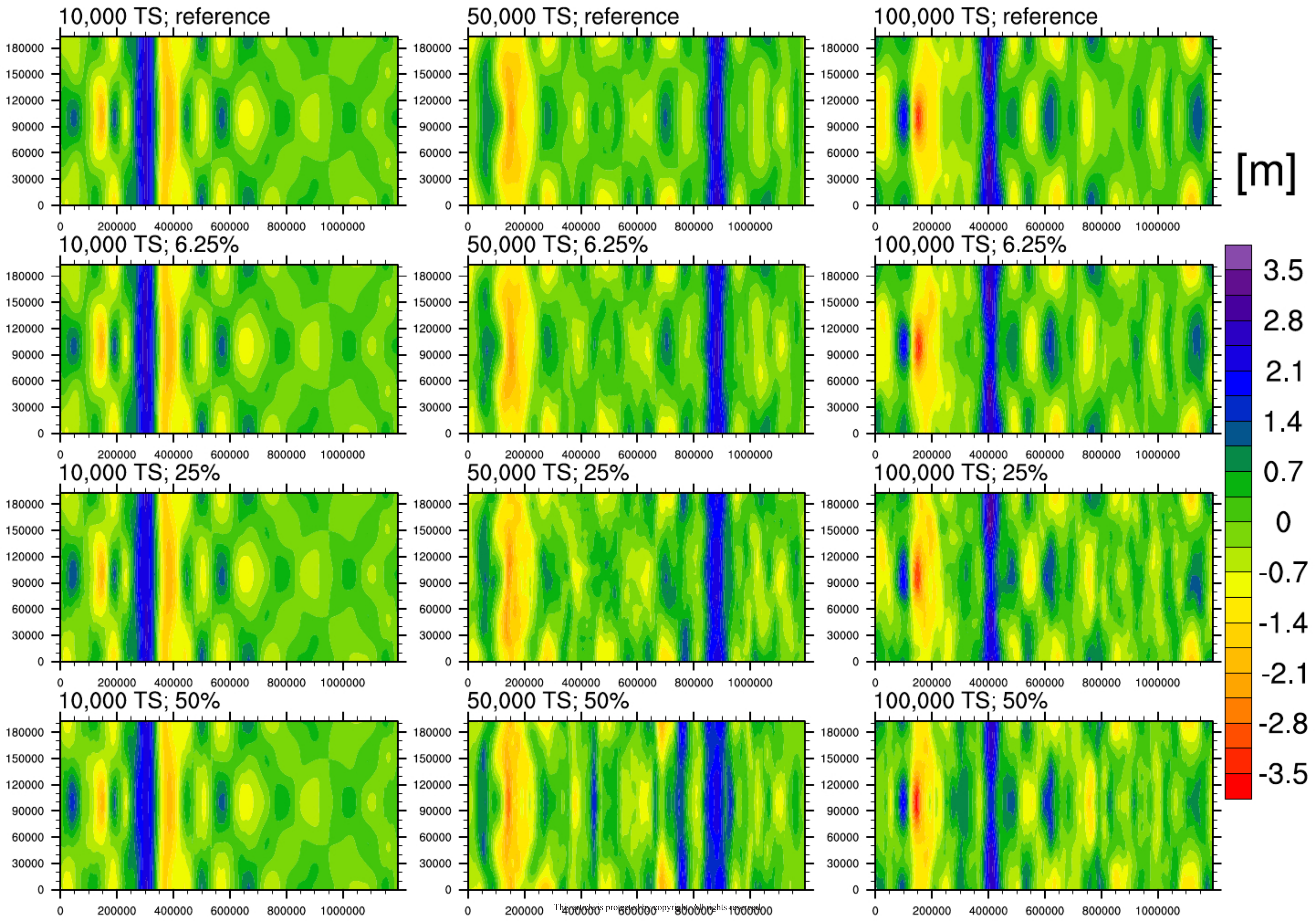


Figure 6.

